DOI. No. 10.1109/MAES.2016.150042

Feature Article:

Design of Telecommand Software for the Mars Orbiter Mission

***Mayur W. Bhilawe, K. K. Raghunandana, Vijaykumar Pujari, H. S. Vasudevamurthy, N. Valarmathi*, Indian Space Research Organization, ISRO Satellite Center, Bangalore, Karnataka, India**

INTRODUCTION

Mars has always interested the human imagination, like no other planet. For ages, humans have been speculating about life on Mars. But, the question still unanswered is whether Mars has a biosphere or ever had an environment in which life could have evolved and been sustained.

The Mars Orbiter [1], [2], is a space probe launched on 5 November 2013 by the Indian Space Research Organization (ISRO) [3]. It is ISRO's first interplanetary mission to planet Mars with an orbiter craft designed to orbit Mars in an elliptical orbit, which has been orbiting Mars since the 24 of September, 2014. With this mission, ISRO has become the fourth space agency to reach Mars, after the Soviet space program, National Aeronautics and Space Administration , and European Space Agency.

The Mars Orbiter Mission (MOM) has many technical challenges considering the critical mission operations and the stringent requirements on propulsion and other bus systems of the spacecraft. It has been configured to carry out limited study of the Martian atmosphere with five payloads [4] finalized by the Advisory Committee on Space Sciences. Figure 1 shows the basic structure of Mars Orbiter spacecraft.

In interplanetary missions where the communication delays are quite high, it takes a significant amount of time to send commands and receive the telemetry. Also, interplanetary missions involve visibility constraints. The round trip delay for a signal in MOM is up to 40 minutes. MOM involves a communication blackout period when the Sun is between the Earth and the Mars. Also, there exist communication white-out periods when the Earth is between the Sun and Mars. Finally, the MOM involves visibility gaps when the spacecraft goes behind Mars. Also, the interplanetary mission involves different phases of mission operations during its life, so it requires flexibility and programmability to manage the mission. Due to visibility constraints and the time delays involved in the mission, autonomy features are developed in the telecommand processor (MTcP) software, which reduces command uplinks and increases the reuse of already uplinked commands.

In this article, the design and development of the MTcP software for the MOM is presented. MTcP software provides autonomy features for thermal management, fault detection, isolation, and reconfiguration (FDIR), differential time tagged (TT) command execution, configurable command block (CCB) execution, event based commanding (EBC), on board time tagged (OBT) command execution, macro time tagged command execution, attitude and orbit control electronics (AOCE) autonomy, 1553B [5]–[7] data/command transfer and telemetry transfer to AOCE, and other subsystems. The software also has features like telemetry (TM) auto-changeover, MTcP auto-changeover, remote programming, telecommand (TC) history transfer to baseband data handling (BDH), and controlled MTcP reset (protection from spurious resets). All these functionalities are elaborated in the subsequent sections. In the MTcP software, some of the functionalities can be combined and used as it provides linking feature. The software is successfully flown in MOM [8], [9]. The picture of Mars Orbiter Spacecraft mounted on a payload adapter in a polar satellite launch vehicle (PSLV) C-25 is shown in Figure 2.

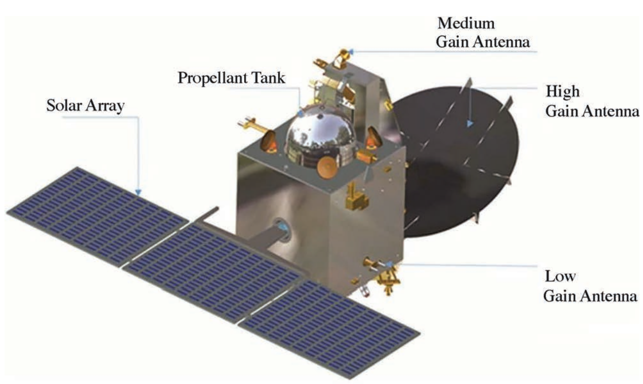
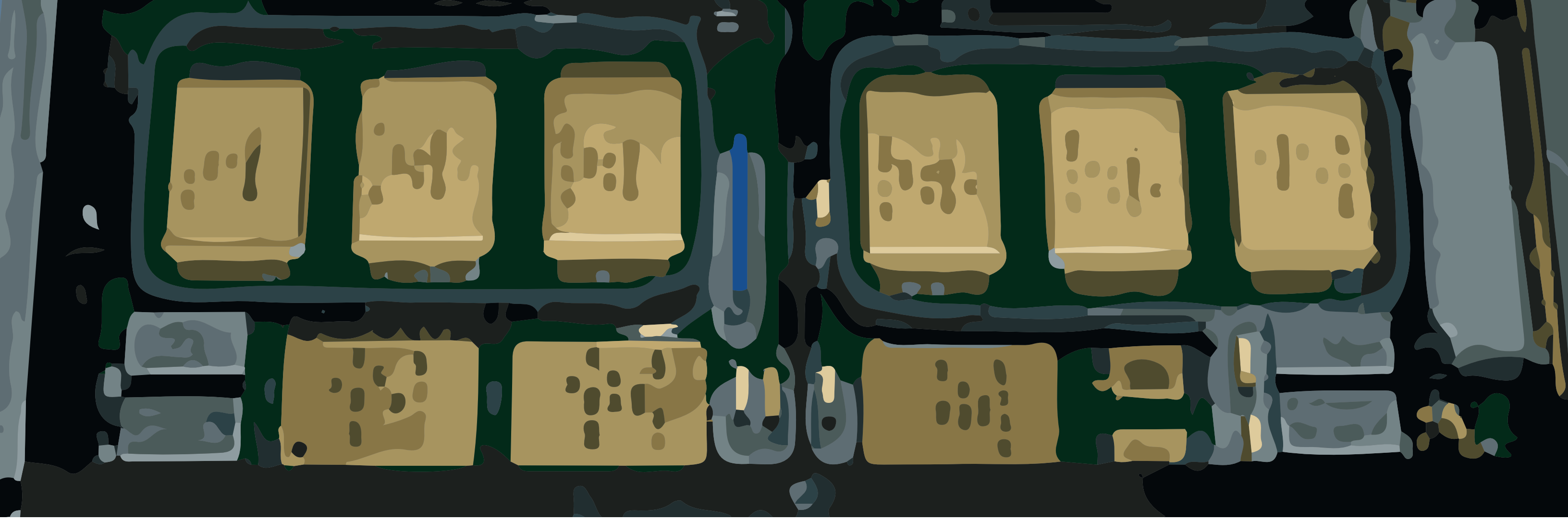


Figure 1.
Mars Orbiter Spacecraft basic structure.

Authors' current address: M. W. Bhilawe, K. K. Raghunandana, V. Pujari, H. S. Vasudevamurthy, N. Valarmathi, Digital Systems Group, ISRO Satellite Center, Indian Space Research Organization, Bangalore, 560017 India, E-mail: (mayur@isac.gov.in).
Manuscript received March 26, 2015, revised August 30, 2015, December 14, 2015, and April 10, 2016, and ready for publication April 10, 2016.
Review handled by E. Blasch.



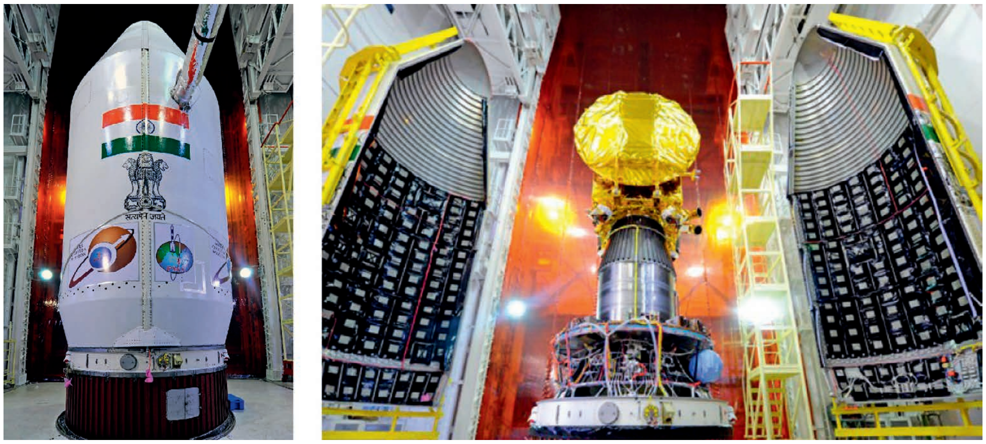


Figure 2.
Mars Orbiter Spacecraft mounted on PSLV C25.

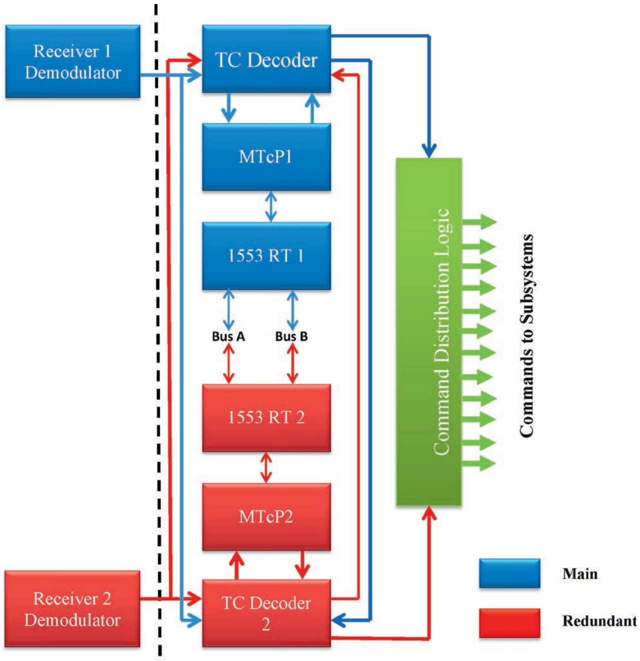


Figure 3.
Telecommand system configuration.

TC HARDWARE DESCRIPTION

The TC subsystem configuration is shown in Figure 3. The TC subsystem is interfaced with S-band receiver. Digital signals from a Phase Shift Keying demodulator [10] are fed to the decoders. The TC subsystem consists of a Consultative Committee for Space Data Systems standard decoder [11], [12], a processor glue logic, a command distribution unit, a MAR31750 microprocessor [13], a memory management unit [14], and a 1553B remote terminal [15].

There is a provision for cross strapping of receivers with decoders. The main and redundant MTcPs are interfaced with main and redundant decoders. Due to cross strapping of receivers with decoders, commands can be sent to decoder 2 from receiver 1 and also decoder 1 can receive commands from receiver 2. Apart from that, decoder 1 can send commands to MTcP 2 and also MTcP 1 can receive commands from decoder 2. There exists a provision to send commands to both the processors simultaneously through any one of the decoders. The simultaneous commanding helps in keeping configuration of both the processors the same. The simultaneous commanding is mainly used in MOM.

The system has a 1553B bus for command/data transfer to AOCE and other subsystems. The AOCE package has the bus controller (BC) for the 1553B bus. The TC provides a remote terminal (RT) interface for AOCE 1553B bus.

The MTcP reads data from the TC decoder and stores, processes, or distributes the command per the pre-defined MTcP modes. The picture of TC flight core card designed for MOM is shown in Figure 4.

SOFTWARE HARDWARE INTERFACE

The hardware-software interaction is through memory mapped Input Outputs [16]. The hardware provides timing reference, telemetry information, and system status for software.

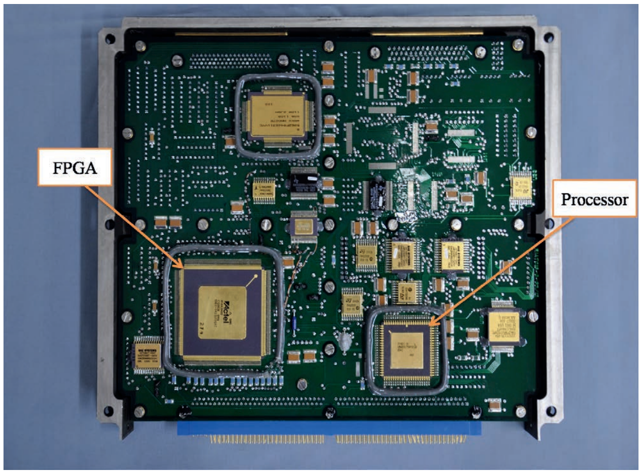


Figure 4.
Telecommand core card.

The Watch Dog Timer is implemented in hardware, which resets the processor periodically if the software failed to reset it within a particular period. The software also gets system identification (ID) to distinguish between main and redundant systems. Therefore, the software is the same in the main and redundant systems but configures itself differently by seeing the system ID.

MTcP SOFTWARE ARCHITECTURE

The on-board software execution is under the control of a Real Time Executive (RTE). The RTE is an infinite loop repeated every major cycle comprising of four minor cycles. The RTE executes the MTcP software functionalities in round robin fashion in the RTE cycle. The MTcP software architecture is general purpose, flexible, and robust. A modified water fall model [17], [18] shown in Figure 5 is used for the development of MTcP software.

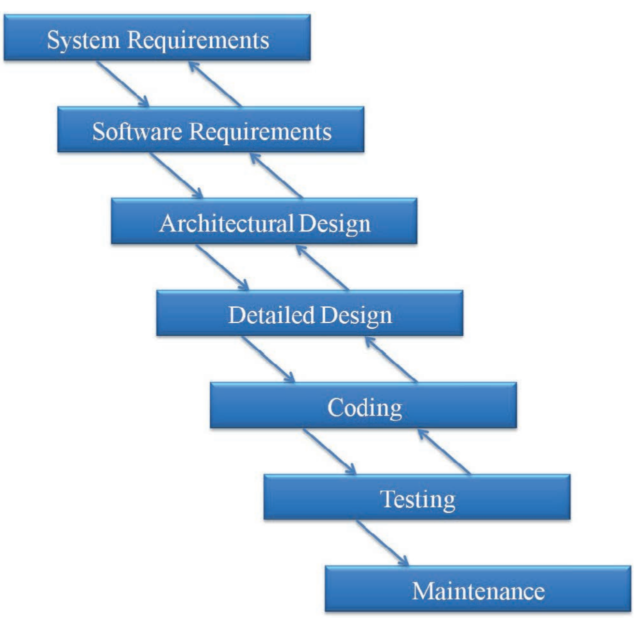


Figure 5.
Modified waterfall model for MTcP software design.

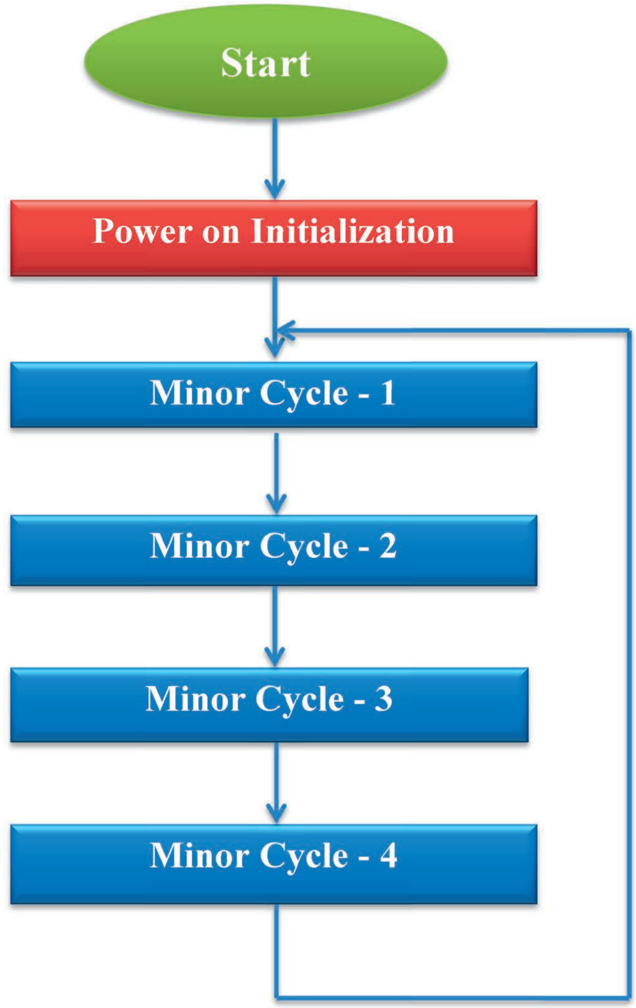


Figure 6.
RTE for MTcP software.

The modified waterfall model is used to maintain the design configuration of the component developments. Each phase has a specific deliverable and a review process. It serves to robustly check operations to validate software changes still resulting in maintained or improved operations.

The MTcP software consists of a power-on initialization routine and a major cycle loop, which runs infinitely. The major cycle loop consists of four minor cycles. Various MTcP functionalities are time sliced/shared within these four minor cycles. The software architecture is shown in Figure 6. Clock driven cyclic scheduling [19] is used with minor cycle time of 16 ms. The minor cycle time is chosen based on software performance under a full load test. Four minor cycles are used as it is sufficient for all the tasks to complete and, also, the generation of the timing reference is easy as it is a power of two.

The various tasks scheduled in different minor cycles are as shown in Table 1. All the tasks within a minor cycle must not take more than 75% of the maximum allotted time according to ISRO Quality Assurance (QA) guidelines. The minor cycle time is optimized to reduce context switches and to satisfy the time deadlines for various functionalities. For example, telemetry frame rate puts a constraint on processing telemetry data within a frame time. EBC may have conditions programmed to check data from a same te-

Table 1.

Minor Cycles Definition			
Minor Cycle 1	**Minor Cycle 2**	**Minor Cycle 3**	**Minor Cycle 4**
Timing management	Timing management	Timing management	Timing management
Command execution logic	Command execution logic	Command execution logic	Command execution logic
Read and decode telecommands	Macro execution check	1553 memory refresh	OBT event decoding and sorting
Thermal management	Self test	EBC condition sampling	OBT event maturity check
TC history transfer to BDH	Snap logic	EBC maturity check	Page display refresh
OBT offset calculation	AOCE event detection	AOCE telemetry acquisition	Macro maturity check
Remote program 1	AOCE event queue execution	TM calibration voltage check	Remote program 4
Memory refresh	EBC condition sampling	Remote program 3	Memory refresh
Global memory refresh	EBC maturity check	Memory refresh	Global memory refresh
--	Remote program 2	Global memory refresh	--
--	Memory refresh	--	--
--	Global memory refresh	--	--

lemetry frame. Therefore, the EBC sampling logic must check all conditions within a telemetry frame time.

The timing management module maintains time for functionalities like TT, OBT, EBCs, etc. It periodically checks for TM health and initiates TM auto-changeover if necessary. The command execution logic decides the priority of command execution. It also handles execution of commands from various functionalities.

MTcP SOFTWARE FUNCTIONS

The MTcP software design is mainly focused on autonomy requirements and flexibility to meet mission requirements during various phases of MOM. Features like TM and MTcP auto-changeover makes it more robust. The software features minimize the ground control intervention in case of any failure as the mission involves significant delays in getting onboard telemetry, as well as visibility constraints.

Features of MTcP software design are explained below. The features are divided as a) EBCs, b) TT Commands, c) FDIR, and d) other features.

EVENT BASED COMMANDING

Telemetry Event Based Commanding

While operating the spacecraft, a general requirement is that when a particular parameter of a subsystem crosses some limit, or if a particular status is set or reset, some specific operation shall be carried out. To meet this requirement, the software monitors onboard telemetry data of the subsystem/sensors and issues commands corresponding to that event. The software includes a set of arithmetic and logical checks (e.g., lesser, greater, all bits high, all bits low, etc.) for the telemetry data. Both event condition and commands are programmable from the ground. The sampling rate of the data is also programmable. A simplified block diagram for telemetry event based commanding logic is shown in Figure 7.

The consistency of telemetry data is monitored and validated to avoid spurious event triggering. Multiple event conditions can also be combined for a particular action.

The EBC feature is a general purpose feature and can be effectively used for normal operations as well as contingency management onboard.

For example, an event can be programmed to switch off the payloads when the battery voltage falls below a particular value. EBCs are also used for battery management in spacecraft. In case of attitude loss, EBCs are used to detect Radio Frequency bit lock and sun presence by rotating the spacecraft.

The same events can be programmed with different conditions and different actions during various phases of the mission. All of the events are provided with individual enable/disable control. The EBC logic is designed to work at all the telemetry data rates.

AOCE Events Autonomy Functions

The AOCE autonomy logic resides in MTcP software. AOCE sends events to MTcP and MTcP takes the action by executing commands corresponding to that event. The AOCE autonomy

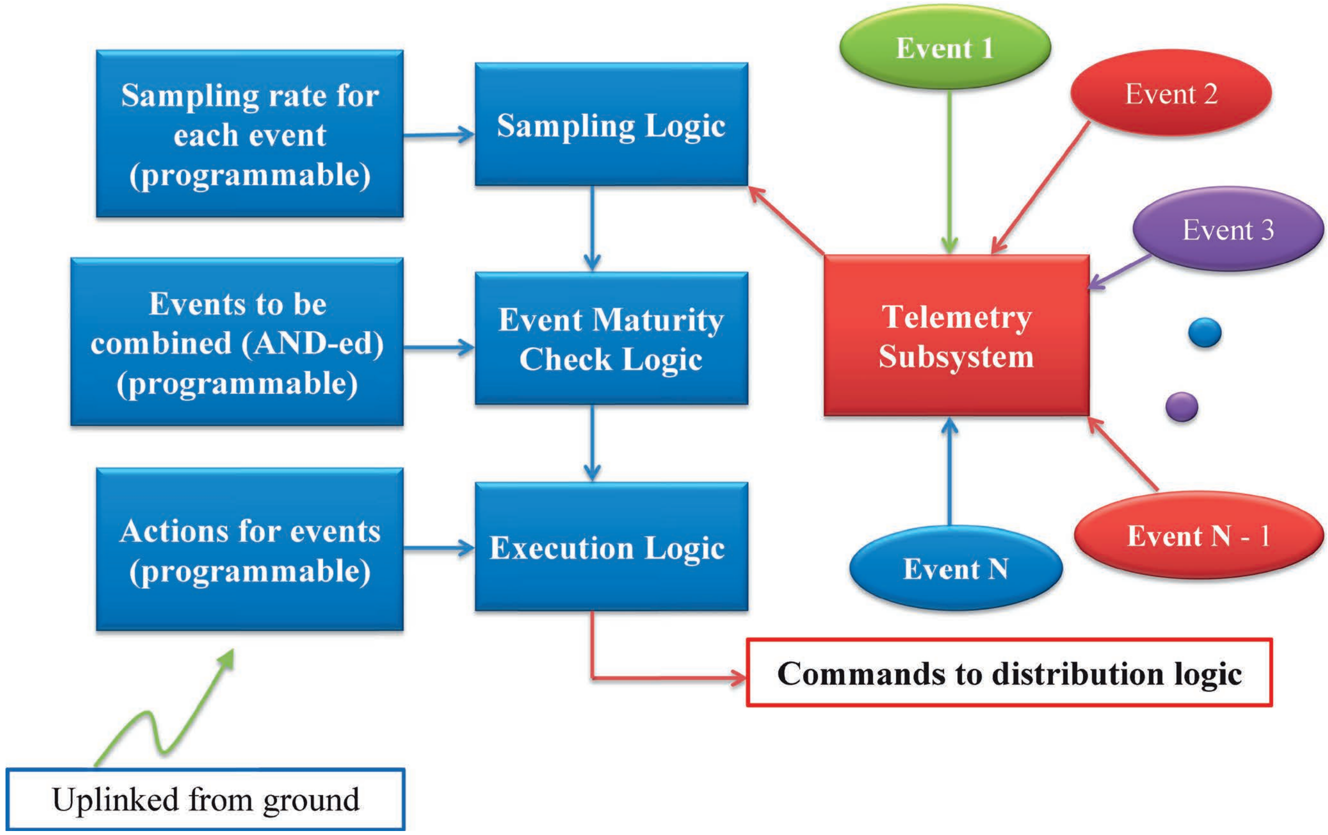


Figure 7.
Telemetry EBC logic block diagram.

commands are differential time tagged commands. Both event and action are programmable.

For example, if one of the star sensors in the loop goes bad, then the event is generated by AOCE to utilize the redundant star sensor in the loop. Therefore, if MTcP gets this particular event, then it executes the commands to bring the redundant star sensor into operation.

Programmable Automatic Thermal Control

Thermal management plays an important role in proper functioning of the subsystems. The subsystems generally have a requirement to maintain the temperature within certain limits.

The software continuously monitors the temperature for all of the subsystems onboard the spacecraft and operates the heaters to maintain spacecraft temperature within certain temperature limits. The temperature limits and sensor information are programmable.

The saturation check logic is also incorporated to take care of the any sensor failure. The programmable automatic thermal control (PATC) logic has an enable/disable feature. Also, each heater has an enable/disable feature.

TIME-TAGGED COMMANDING

Differential Time Tagged Command Execution

The differential TT commands belong to the class of time tagged commands. The TT commands are useful when a particular sequence of action and delay is required between the commands.

Each TT command has a differential delay associated with it. Both command and delay are programmable from the ground station.

The TT command execution is shown in Figure 8. When the TT execution start is given, a certain number (N) of commands are executed one by one according to differential delays associated with them. The TT execution start consists of start command number and number of commands to be executed.

Snap Logic

The snap signal gets generated when the spacecraft gets separated from the launch vehicle. After the snap signal is detected, some predefined operations have to be done. These operations are loaded in the TT stack at the launch pad. The trigger to these commands is through the snap signal. The snap logic handles the snap signal and takes the proper action after the spacecraft is separated from the launch vehicle.

For example, a snap sequence consists of actions like solar panel deployment, thruster selection, etc. When the spacecraft gets separated from launch vehicle, the snap sequence is loaded as TT commands get activated.

Macro

Macro is a block of differential TT commands which can run in parallel independently. The commands and delays are programmable. The Macro TT is slightly different from the conventional differential TT as it has two level controls. The Macro TT becomes active when it is enabled and initiated.

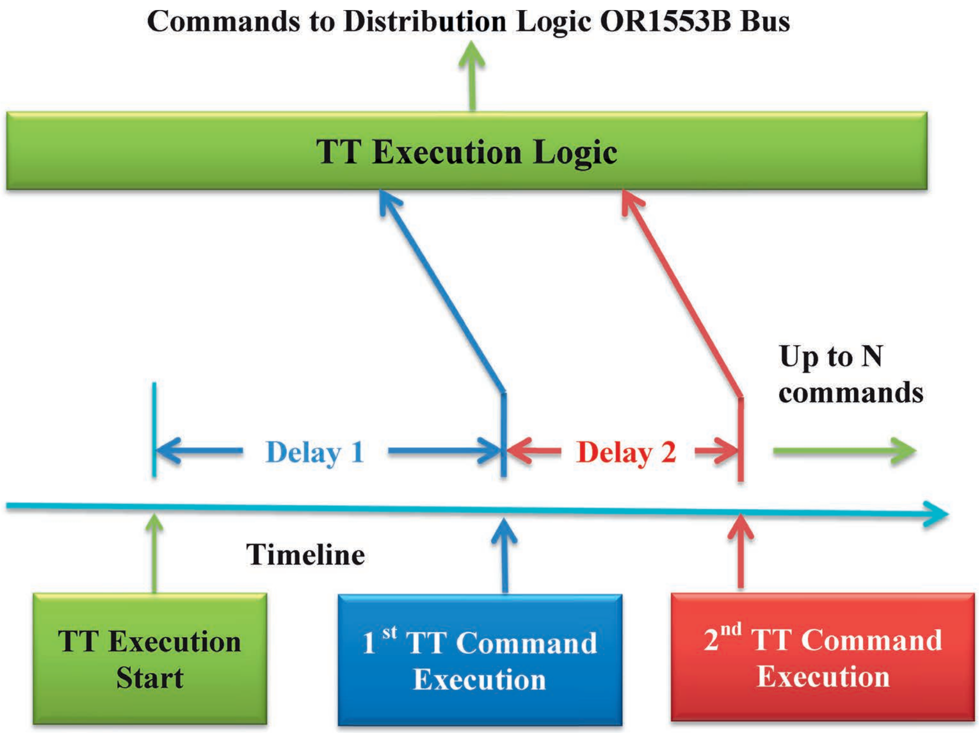


Figure 8.
Differential TT command execution.

OBT commands are extremely useful when the spacecraft is not visible or in scenarios where direct commanding is not possible. For example, OBT commands are used for Liquid Apogee Motor (LAM) firing in case of visibility constraints.

The two level control is an extremely useful feature in design and planning of autonomy. For example, in case of Travelling Wave Tube Amplifier (TWTA) autonomy, if the TWTA needs to be switched on at a particular OBT but only when the battery voltage is above a particular value, then the OBT event can enable a Macro TT and an EBC (checking for battery voltage) can initiate it.

Also, the macros are useful for payload operations. The Mars color camera has been operated by OBT commands triggering macros to take pictures of the Mars.

On Board Time Tagged Command Execution

The OBT tagged commands also belong to the class of time tagged commands. The onboard timer reference is from telemetry system, which is running continuously from system power on till end of the mission. The OBT commands are executed when the time running onboard matches with the OBT associated with the command.

The uplinked commands are sorted according to the OBT associated with them. The OBT logic compares the OBT of a command having the least OBT stamp with the onboard running OBT. The OBT command is executed when the on board running OBT matches with the command OBT.

The OBT software block diagram is shown in Figure 9. The uplinked commands are first validated by OBT validation logic. The validated commands are then processed by OBT sorting logic. OBT processing logic computes OBT difference between main and redundant TM OBTs to take care of telemetry auto-changeover. The computed difference is used to correct OBT after TM auto-changeover.

The software also has the features like OBT command deletion and OBT stack reset. The OBT stack reset deletes all the OBT commands on board.

FAULT DETECTION, ISOLATION, AND RECONFIGURATION

FDIR is important in this mission. The ground station will come to know any problem onboard the spacecraft after quite a large time as the distances involved are huge. Therefore, an effective mechanism is required to identify the faults, which can cause mission loss. As we have seen, the general purpose functions in MTcP software can be programmed to take care of failures in the other subsystems. But what will happen if the telecommand subsystem itself has a problem is addressed in this section.

Telemetry Auto-Changeover

Both main and redundant telemetry are available onboard. The MTcP works on selected telemetry; if the selected telemetry onboard goes into a problem state, then telemetry auto-changeover happens. For example, the parameters for TM auto-changeover are frame pulse failure, OBT errors, calibration voltage errors, TM frame cyclic redundancy check errors, etc.

As both main and redundant telemetry are asynchronous, features like PATC, EBC, and OBT dependent on TM data are designed to function properly after TM auto-changeover.

For example, in the case of OBT, generally the ground station selects one of the telemetry (let's say TM1) and uplinks all the OBT commands with respect to the OBT in the selected telemetry. If the selected telemetry goes into problem, then telemetry auto-changeover happens. As the TM1 OBT reference is lost, the MTcP derives the TM1 OBT from TM2 OBT after TM auto-changeover as the uplinked commands have TM1 OBT tagged with them. Therefore, the OBT modules work even after the TM auto-changeover happens.

MTcP Auto-Changeover

The MTcP auto-changeover makes the TC subsystem robust. If there is a problem in the selected TC subsystem, then the auto-changeover happens to the non-selected/redundant subsystem and MTcP operation continues. So the system can withstand a TC subsystem failure until the ground station intervenes.

The basic idea used for MTcP auto-changeover is as follows. The commands are uplinked to both the processors simultaneously

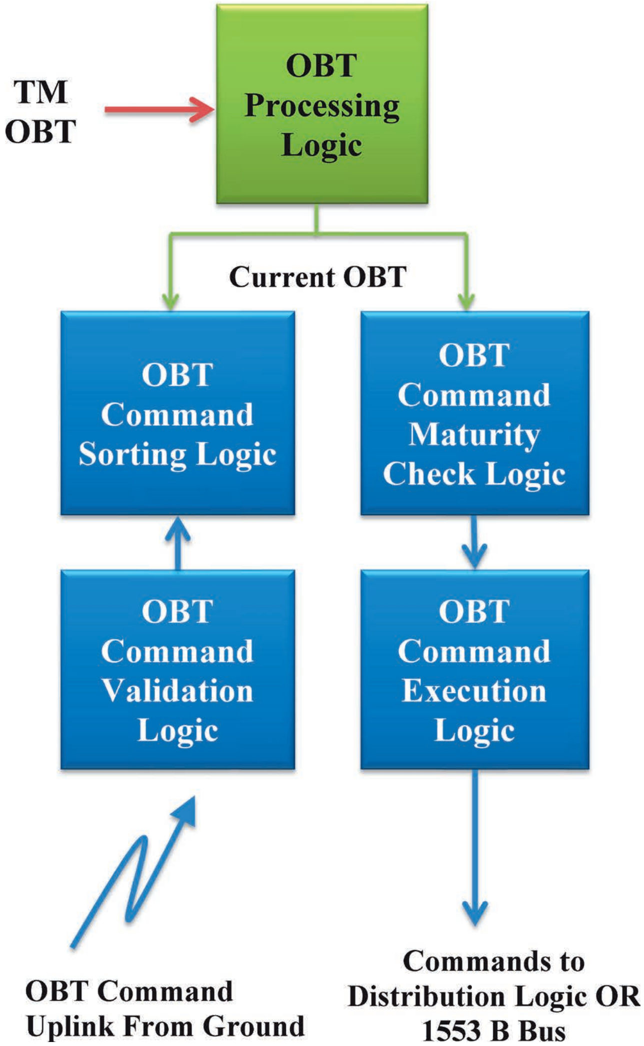


Figure 9.
OBT logic block diagram.

from the ground station so that their configuration and database remains synchronized if there is an uplink from the ground station. The changes done by the enabled/main processor in configuration onboard are transferred to a disabled/redundant processor so that their configuration remains the same after onboard actions. The functionalities running in a redundant processor monitor the same functionalities running in main processor and update their configuration by looking at what is happening in main processor. This helps the functionalities running in a redundant processor to resume operation from the same point where main processor failed.

For example, the OBT module running in main MTcP executes the commands when the OBT matures, but the OBT module running in redundant processor deletes the OBT events whose time has matured instead of executing the commands. Therefore, both main and redundant OBT modules remain in synch. After MTcP auto-changeover happens, the OBT module running in redundant MTcP starts executing the commands and the OBT operation is as expected until the ground station intervenes.

The functional diagram of TC autonomy is shown in Figure 10. Two kinds of failure detection are local and remote. For detecting remote system failure, the heart beat signal is exchanged between the local and remote system. The heart beat signal tells about the health of that system to the other system. If there is a problem in the heart beat signal of the local system in a time window, then the remote system automatically takes over.

In local system failure, the local hardware detects it and initiates the changeover.

OTHER FEATURES

Configurable Command Blocks

CCB is a block of commands. These commands don't have any delay/time associated with them. The uplinked commands are validated and stored onboard. The CCB is specified by a block number.

The CCB is a set of commands which perform a definite operation when executed. The CCB can be reused again and again.

For example, CCBs are used for storing load shedding commands and also configuration commands, which can be triggered whenever required.

Real-Time Commands

Real-time commands have the highest priority among all the commands. These commands are executed immediately after validating the uplinked command.

For example, liquid engine burn commands are given in real time if the spacecraft is visible during the earth bound phase.

Command/Data Transfer on 1553 Bus

The TC system has a RT on the AOCE 1553B bus. All the 1553B commands/data are sent on this bus. The 1553B commands/data can be sent to AOCE BC and other RTs on the bus. The software supports real time, TT, OBT, and CCB commands through the 1553B bus.

The software also transfers telemetry parameters to AOCE over the 1553B bus. All telemetry data are programmable.

The MTcP software sends an executed telecommand history to a baseband data handling package on the 1553B bus. Therefore, the telecommand history is available in playback data. The executed telecommand history is very useful for post-operation analysis.

The main MTcP software also transfers data to redundant MTcP software in order to keep redundant MTcP software configuration the same as the main MTcP software. This configuration synchronizes redundant MTcP to take over when MTcP auto-changeover happens.

Remote Programming

The remote programming feature enables the MTcP software to accommodate new requirements. If any new software modules are required to meet the new requirements, then new software modules can be uplinked through ground commands.

The remote program is provided with an enable/disable feature.

Linking Features

The real strength of the MTcP software is the ability to combine various MTcP features to perform an operation. It provides flexibility to

the mission team for planning complicated, autonomous operations.

The linking feature also enables the reuse of already uplinked commands by triggering them again and again through EBCs, OBT, AOCE autonomy events, etc.

For example, EBC triggering macro TT commands are used for Traveling Wave Tube Amplifier (TWTA) autonomy. The EBC detects the main TWTA failure and triggers macro TT to reconfigure the transmission chain to use redundant TWTA.

Diagnostic Features

The processor status words reported in telemetry indicate the telecommand system health. As an interplanetary mission involves significant delays in getting telemetry, it takes a lot of time to dump the uplinked data to ground stations. Therefore, a checksum dump feature is provided for getting the checksum of all the database and commands stored onboard.

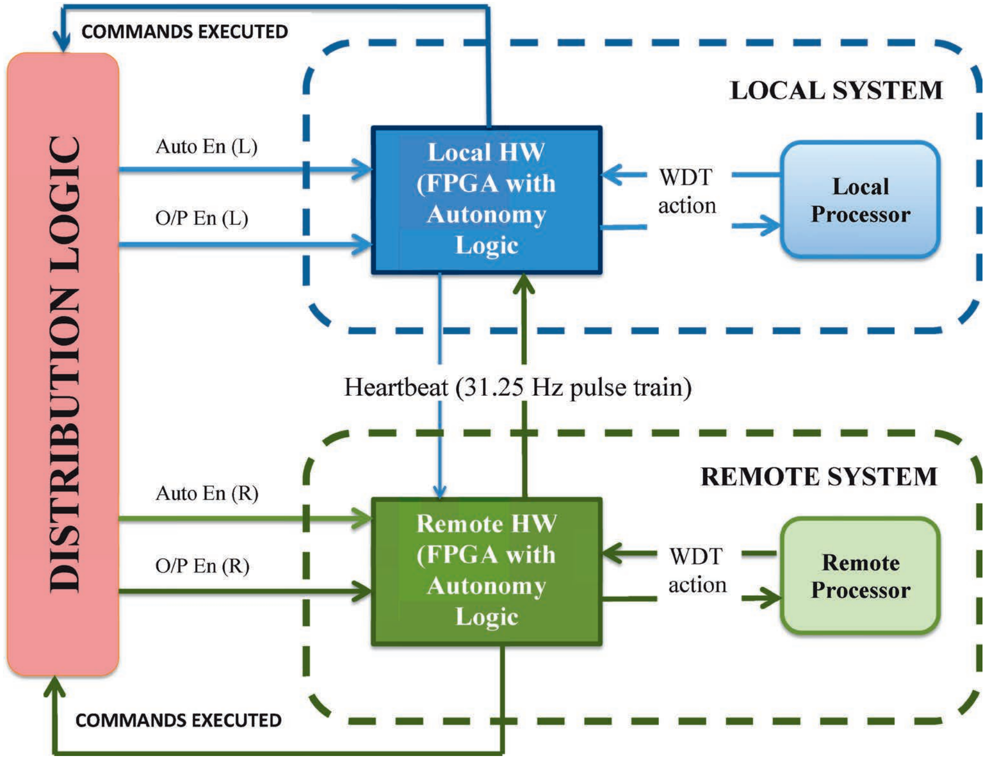


Figure 10.
TC Autonomy functional diagram.

TESTING AND EVALUATION

Telemetry-telecommand (TMTC) package along with TC core card and distribution card is shown in Figure 11.

Extensive unit level and system level testing [20], [21] was carried out for the MTcP software. The unit level and system level testing was carried out to verify the following:

- Hardware software interface
  - Uplinked command buffer update
  - Uplinked command counter update
  - Snap and Safe Mode Operation
  - Pulse/level/data commands
- Command decoding of all types of commands
  - Decoding of proper sequence of uplinked commands
  - Handling of improper sequence of uplinked commands
  - Boundary checks
- Command validation logics
- Checksum of data/command and actual data/command verification stored onboard
- Execution of commands
  - TC History to BDH
- Sorting algorithm for OBT
- Priority of execution
- TM auto-changeover
- MTcP auto-changeover

In addition to the above testing, code walk through, design, and database verification was done. All the mission plans were tested in the setup shown in Figure 12 before actually sending them to the spacecraft. The setup consists of AOCE and TC packages equipped

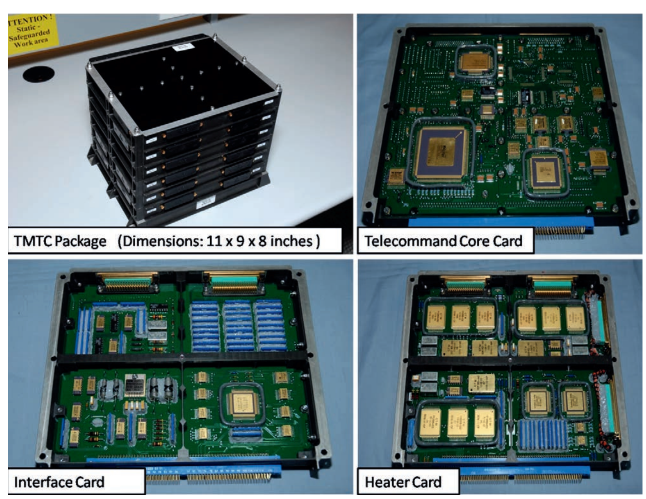


Figure 11.
TMTC package in MOM.

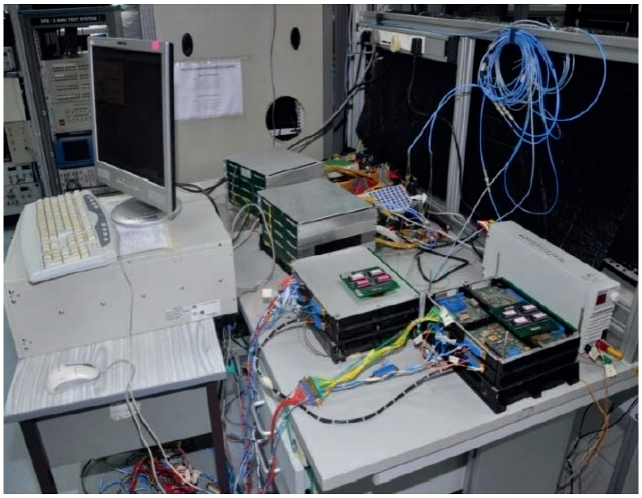


Figure 12.
AOCE-TC setup for MOM.

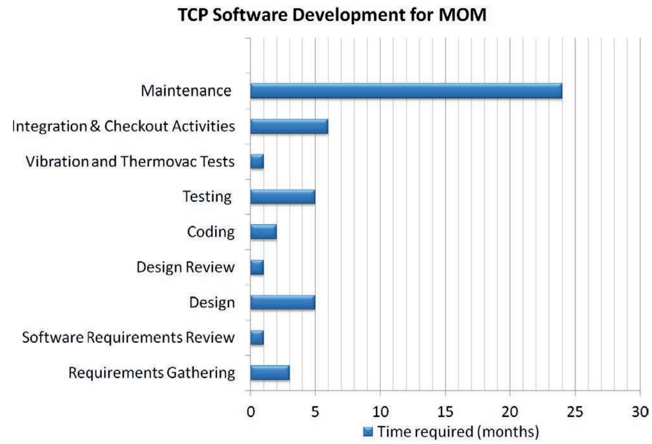


Figure 13.
Time required for MTcP software development.

with a test station which is simulating telemetry and other interfaces.

RESULTS

Time required for various activities in MTcP software development is shown in Figure 13. The entire software was developed in one and a half years. The exact path to Mars is known to very few people. Therefore, it was very difficult to finalize the requirements. Apart from that, testing of auto-changeover under various mission scenarios was challenging. The fault injection was a difficult part of testing when most of the interfaces are asynchronous in nature.

Table 2 gives the time line for the development of MTcP software in MOM. The MTcP software was developed with high priority, as the launch window for launching the Mars Orbiter spacecraft was in October–November 2013.

Table 2.

Time Line for MTcP Software Development

Development activities	Date started (dd-mm-yyyy)	Date completed (dd-mm-yyyy)	Number of days
MTcP Software requirements specification review	01-03-2012	05-04-2012	36
MTcP software design	06-04-2012	02-09-2012	150
MTcP software design review	03-09-2012	30-09-2012	28
MTcP software coding	01-10-2012	15-11-2012	45
MTcP software testing by designers	16-11-2012	10-12-2012	26
Incremental design review for design changes (based on testing by designers)	11-12-2012	15-12-2012	5
Incremental testing for the design changes by designers	16-12-2012	01-01-2013	17
Testing of software by quality assurance division	02-12-2012	12-03-2013	101
Incremental design review for changes in design (based on QA observations)	06-03-2013	12-03-2013	7
Incremental testing of software by quality assurance division	13-03-2013	20-03-2013	8
Clearance for Programmable Read-Only Memory fusing by Subsystem Review Board	21-03-2013	21-03-2013	1
MTcP software fused	22-03-2013	22-03-2013	1

LESSONS LEARNED

MOM was challenging as the mission plans used to change depending on the current situation and inputs from the review committee. There were few observations related to mission operations. The observations and the lessons learned are listed below.

During one of the maneuvers, the LAM could not give the expected performance. The spacecraft could not achieve the expected delta velocity due to under-performance of the LAM. After the analysis of the observation, it was found that the mission plan executed on that day was never tested in checkout. Therefore, it is better to avoid the operations that are never tested at ground checkout.

The Mars Orbiter has many autonomy logics onboard. One of the logics, TWTA autonomy, changes the transmission chain automatically if the current transmission chain lands into a problem to ensure continuous telemetry transmission to ground station. When the logic was enabled, complete telemetry was lost as the TM channel programmed for detecting TWTA failure was wrong and the logic misfired. The TWTA autonomy logic could not change the transmission chain as the actions (commands) uplinked were not correct. The review committee advised to simulate the commands on the setup shown in Figure 12 before sending the commands to the spacecraft to avoid such mistakes.

We also faced a few problems in thermo-vacuum and interface testing. The observations and lessons learned during the development phase are listed below.

During the development, one of the Low Voltage Transistor-Transistor Logic Bus Hold (LVTH) [22] inputs was floating (inputs that are not pulled up or down) and LVTH output was a part of bus arbitration logic. At room temperature, the system passed all of the tests. When the system was switched off and on at low temperature (-18 degrees Celsius), the bus arbitration logic misbehaved; it was due to LVTH output which toggled its state at that temperature. The processor could not initialize the software properly due to the failure of bus arbitration logic and in turn behaved like it was hanged. Therefore, it is better to avoid floating inputs. These kinds of observations are very difficult to analyze as it takes time to bring the thermo-vacuum chamber to -18 degrees Celsius if one wants to repeat the observation.

Interface tests with other subsystems are very important. The MTcP transfers the executed commands history to BDH over 1553B bus. During the transfer, BDH receives the valid number of words followed by the data (TC history) in a sub-address and it stores the TC history. The number of words is a 5-bit field. 1–31 is represented as 1–31 and 32 is represented as 0. This representation is same as the number of words field in the command word of 1553B protocol. The BDH software was expecting number of words to be 32 when the data has 32 valid words but the MTcP software was giving it as 0 as it was a 5-bit field. Therefore, when the valid number of words was 32, BDH was missing the data which we observed during interface test.

There were no observations in the software performance after it was delivered for flight. All the functionalities worked per the defined requirements. The mission was perfect from the software point of view.

CONCLUSION

The MTcP software design concepts presented in this article are successfully flown in MOM. The software is working well onboard the Mars Orbiter. The software design presented in this article is simple, general purpose, and database driven. The design is robust and can withstand the main system failure by changing over to a redundant system automatically. The configurable features provided can be used to achieve autonomy in spacecraft. Due to programmability provided in the software, the same features can be used for different purposes in different phases of a mission. ◆

ACKNOWLEDGMENTS

The authors would like to thank and acknowledge the support and encouragement received from Mr. E. Vasantha, Deputy Director, Digital and Control Area (DCA) and Dr. Shivakumar S. K., Director ISRO Satellite center (ISAC), Bangalore. They would also like to acknowledge the contribution of the hardware design team of telecommand for their contribution towards understanding the hardware interfaces and the contribution of test system team for designing a new, advanced test station for Mars Orbiter Mission.

REFERENCES

[1] Mars Orbiter Mission. Nov. 5, 2013. [Online]. Available: http://www.isro.gov.in/pslv-c25-mars-orbiter-mission, last access Aug. 5, 2015.

[2] Mars Orbiter Mission. Aug. 4, 2015. [Online]. Available: https://en.wikipedia.org/wiki/Mars_Orbiter_Mission, last access Aug. 5, 2015.

[3] Kramer, M. Liftoff! India's first mars probe launches toward the red planet. Nov. 5, 2013. [Online]. Available: http://www.space.com/23464-india-launches-mars-orbiter-mission.html, last access Aug. 5, 2015.

[4] Payloads. Nov. 5, 2013. [Online]. Available: http://www.isro.gov.in/pslv-c25-mars-orbiter-mission/payloads, last access Aug. 5, 2015.

[5] Condor Engineering. MIL-STD-1553 tutorial. *Condor Engineering*. Oct. 3, 2000. [Online]. Available: http://microsat.sm.bmstu.ru/e-library/military%20standatds/MIL-STD-1553Tut.pdf, last access Dec. 3, 2014.

[6] MIL-STD-1553B: Digital time division command/response multiplex data bus. United States Department of Defense, Sept. 1978.

[7] Data Device Corporation. MIL-STD-1553 designer's guide sixth edition. Data Device Corporation, Aug. 12, 2003. [Online]. Available: http://www.ddc-web.com/Documents/dguidehg.pdf, last access Aug. 1, 2015.

[8] ISRO. Mars Orbiter spacecraft successfully inserted into Mars orbit. isro.org, Sept. 24, 2014. [Online]. Available: http://www.isro.org/update/24-sep-2014/mars-orbiter-spacecraft-successfully-inserted-mars-orbit, last access Aug. 5, 2015.

[9] Mars Orbiter mission successfully enters Red Planet orbit. *India Today*, Sept. 24, 2014. [Online]. Available: http://indiatoday.intoday.in/story/mars-mission-mars-orbit-insertion-mom-isro-india-m-annadurai-baylalu-red-planet/1/384723.html, last access Aug. 5, 2015.

[10] Kulkarni, S., Sharma, S., Pujari, V., Lakshminarsimhan, P., and Seshaiah, R. A Costas loop PSK demodulator with in phase/mid phase bit

synchronizer—a proto design, simulation and test results. In *Proceedings of the IETE 36th Midterm Symposium on Emerging and Futuristic Communication System*, Bangalore, Mar. 2005, 252–258.

[11] CCSDS. TC synchronization and channel coding. CCSDS 231.0-B-1, Sept. 2003.

[12] CCSDS. TC space data link protocol. CCSDS 232.0-B-1, Sept. 2003.

[13] Dynex Semiconductor. MA31750 high performance MIL-STD-1750 Microprocessor. MA31750 datasheet, Nov. 2000.

[14] Dynex Semiconductor. Memory management and block protection unit. MA31751 datasheet, Nov. 2000.

[15] *Enhanced Summit Family Product Handbook*. Colorado Springs, CO: UTMC Micro-Electronic Systems Inc., Oct. 1999.

[16] Simon, D. E. Advanced hardware fundamentals. In *An Embedded Software Primer.* Delhi, India: Pearson Education, 2005, pp. 70–72.

[17] Jalote, P. Software processes. In *Integrated Approach to Software Engineering* (2nd ed.). New Delhi, India: Narosa Publishing House, 2005, pp. 23–71.

[18] Pressman, R. S. Process models. In *Software Engineering–A Practitioner's Approach* (7th ed.). New Delhi, India: McGraw-Hill, 2010, pp. 30–64.

[19] Kharagpur, IIT. Module 6 embedded system software version 2. NPTEL. [Online]. Available: http://www.nptel.ac.in/courses/108105057/Pdf/Lesson-29.pdf, last access Mar. 3, 2015.

[20] Raghavaiah, V., et al. Electromagnetic compatibility of Mars Orbiter Mission spacecraft. In *Proceedings of the IEEE International Microwave and RF Conference,* Bangalore, 2014, 209–212.

[21] Shankar S, S., et al. Mission critical software test philosophy a SILS based approach in Indian Mars Orbiter mission. In *Proceedings of the International Conference on* Contemporary Computing and Informatics (IC3I), Mysore, India, 2014, 414–419.

[22] Texas Instruments. SN54LVTH162244, SN74LVTH162244 3.3V ABT 16 bit buffers/drivers with 3-state outputs. SN54LVTH162244 datasheet, Sept. 2006.